

## Controller Area Network

### или, как более привычно звучит для автомобильной диагностики — CAN шина

---

- \* *Что такое CAN?*
- \* *Взаимосвязь открытых систем (Open System Interconnection (OSI))*
- \* *Controller Area Network (CAN)*
- \* *Основные принципы CAN*
- \* *Как выглядит CAN шина на примере автомобилей произведённых в Японии*

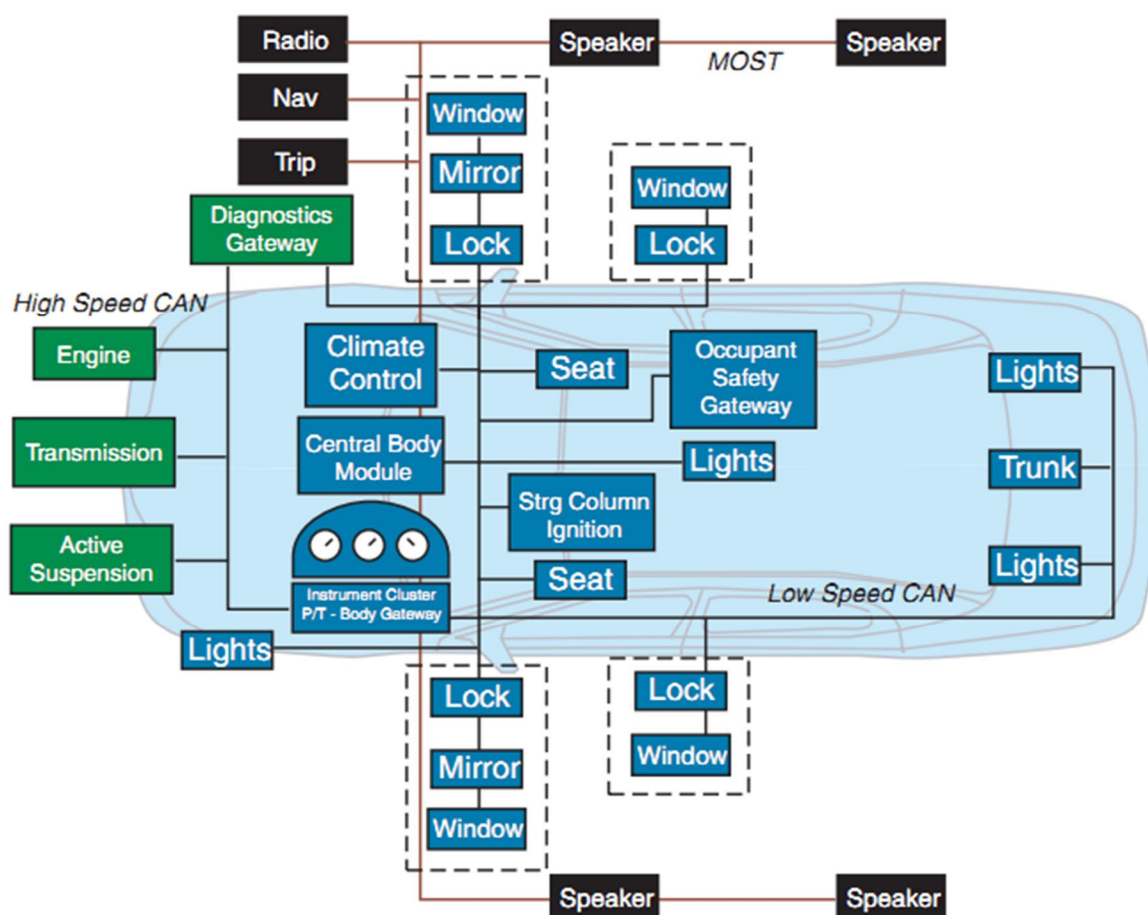
Парк автомобилей на наших улицах стремительно омолаживается и вместе с этим приходится осваивать и решать новые задачи связанные с диагностикой и ремонтом. Всё чаще и чаще сталкиваешься в своей повседневной работе с проблемами коммуникации между различными бортовыми системами автомобиля. Если ещё несколько лет назад приезжающие на диагностику автомобили с ошибками по **CAN шине** (первый символ в классификации диагностического кода неисправности — **U**) были редкими гостями, то сейчас это практически повседневная практика. Информация на эту тему в принципе доступна и её достаточно много, даже очень много - что с одной стороны хорошо, а с другой представляет собой определённую сложность в поиске необходимой информации. Этой статьёй хотелось бы в первую очередь дать общее представление о системе **CAN (Controller Area Network)** тем, кто только начинает с ней знакомство, и тем, кто желает в этом поглубже разобраться.

### Что такое CAN?

**Controller Area Network** — это понятие вошло в обиход после того, как в начале 1980-х годов в **Robert Bosch GmbH** разработали стандарт промышленной сети, ориентированный прежде всего на объединение в единую сеть различных исполнительных устройств и датчиков. Одно из первых внедрений в автомобильной промышленности было осуществлено на нескольких моделях автомобилей **Mercedes-Benz** в 1992 году. До этого момента электронное управление исполнительными функциями строилось по системе - один блок управления принимал электронные сигналы с различных датчиков и после их обработки посылал соответствующие команды на исполнительные устройства (такие как бензонасос, форсунки, катушки зажигания и прочие...). Увеличение объёма функций управления автомобилем, передаваемое электронике, привело к появлению таких дополнительных систем как ABS, SRS, AT, Immobiliser и

других... Совмещение этих функций в одном ЭБУ привело бы к его громоздкости и чрезмерной сложности, а так же к потере надёжности, когда выход из строя одной системы мог бы привести к потере управляемости всего автомобиля. Поэтому автопроизводители пошли по пути разделения функций управления и выделения всех систем в отдельные блоки. А для того, чтобы увязать все системы в единое целое для решения общих задач управления автомобилем, на помощь пришёл коммуникационный стандарт **CAN** от **Robert Bosch GmbH** и это всё шире и шире стало применяться в автомобилестроении. На сегодняшний день практически каждый новый автомобиль оснащён этой системой.

Всё в принципе просто и понятно, но как устроена **CAN шина** и на чём основывается принцип её работы? Вот один из примеров взаимосвязи электронных блоков управления и устройств завязанных в единую бортовую коммуникационную сеть автомобиля, - **рис. 1**



Здесь мы рассматриваем только блоки, связанные в проводную сеть, но в автомобилях 21 века находит всё большее применение и беспроводная передача

\*\*\*\*\*  
[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

информации. К примеру, система навигации, слежение за местонахождением автомобиля (защита от угона), контроль за давлением в шинах, удалённая диагностика и многие другие. В ближайшем будущем можно ожидать, что слияние воедино в бортовой сети автомобилей внутренних и внешних информационных потоков выведет управление транспортным средством на новый уровень безопасности и комфорта прежде всего в таких направлениях, как отображение информации предупреждения об опасных ситуациях на дорогах и даже активного смягчения последствий возможных столкновений автомобилей, а так же более рационального распределения транспортных потоков.

### Немного предыстории — Взаимосвязь открытых систем (Open System Interconnection (OSI)).

Это очевидно, что если два или более микропроцессора взаимосвязаны в одну систему, то должен использоваться стандартный протокол который определяет, каким образом данные должны быть переданы между сетевыми блоками. Наиболее распространенным примером такого протокола является **TCP/IP (Transmission Control Protocol / Internet Protocol)**, который используется для подключения хостингов в сети Интернет. Предшественником **TCP/IP** был протокол — **Open System Interconnection (OSI)**. Этот протокол был разработан в 1982 году Международным бюро по стандартизации **International Organization for Standardization (ISO 7498-1:1994 (E))**. **OSI** протокол иногда называют как «семиуровневая» модель, поскольку он состоит из семи независимых элементов, которые определяют требования к взаимосвязи на различных уровнях взаимодействия.

#### Вот эти семь уровней:

- 1) Уровень приложений (**Application Layer**) — этот уровень определяет какие приложения (программы) имеют доступ к сети. Например — электронная почта, передача файлов, терминалы удалённого доступа и веб-браузеры.
- 2) Уровень представления данных (**Presentation Layer**) — этот уровень определяет такие моменты, как стандарты сжатия данных и их шифрования.
- 3) Уровень передачи данных (**Transport Layer**) — этот уровень обеспечивает стандарты передачи данных между адресатами, осуществляет контроль ошибок и безопасности.
- 4) Сетевой уровень (**Network Layer**) — этот уровень отвечает за вопросы маршрутизации сетевого трафика данных.
- 5) Уровень каналов связи (**Data Link Layer**) — этот уровень обеспечивает синхронизацию передачи данных и контроль ошибок.

\*\*\*\*\*

[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

6) Уровень контроля за сеансами связи (**Session Layer**) — этот уровень обеспечивает стандартизацию начала и завершения сеансов связи между различными приложениями и сетевыми блоками.

7) Физический уровень (**Physical Layer**) — этот уровень определяет стандарты физических характеристик устройств в сети, в том числе типы соединений и разъёмов, электрические характеристики кабелей, уровня напряжения, силы тока и тд.

Но задачи, решаемые протоколом **OSI** не в полной мере отвечали нуждам автомобильной электроники, и как следствие этого, инженерами **Robert Bosch GmbH** был разработан, в развитие протокола **OSI**, специальный протокол **CAN**, который определял стандарты физического и канального уровней модели **OSI** в кремнии для осуществления последовательной передачи информации между двумя или более устройствами.

### Controller Area Network (CAN)

**CAN** был разработан **Robert Bosch GmbH** для автомобильной промышленности в начале 1980-х годов и официально публично выпущен в пользование в 1986 году. Эта разработка **CAN** от **Bosch** была принята в качестве стандарта **ISO (ISO 11898)**, в 1993 переименована в **CAN 2.0A**, и расширена в 1995 году, чтобы позволить идентифицировать большее количество сетевых устройств в **CAN 2.0B**. Как правило, **CAN шина** соединяет в сеть модули (или узлы), используя два провода, витая пара. Многие компании и не только автомобильные, внедряют **CAN** протокол в свои разработки для взаимосвязи различных электронно-управляемых устройств. В неофициальной классификации устройства связанные протоколом **CAN** и имеющие процессоры серии **MPC 5xx**, называются **TouCAN** модули; имеющие процессоры серии **MPC 55xx** называются **FlexCAN** модули. **CAN** - последовательный, мульти-отправляющий, многоадресный протокол, это означает, что, когда шина свободна, любой узел, может отправить сообщение (мульти-отправляющее устройство), и все узлы могут получить и отреагировать на сообщение (многоадресно передано). Узел, который инициирует сообщение, называют передатчиком, любой узел не отправляющий сообщение называют получателем. Всем сообщениям присвоены статические приоритеты, передающий узел остаётся передатчиком до тех пор пока шина не станет неактивной или пока в сети не появилось сообщение от другого узла с более высоким приоритетом, процесс который определяет приоритет сообщений и называется — арбитраж. Сообщение по **CAN шине** может содержать до 8 байтов данных. Идентификатор сообщения описывает контент данных и используется получающими узлами для определения места назначения в сети (другими словами — адресата, узел которому это сообщение адресовано). В

\*\*\*\*\*

[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

коротких сетях ( $\leq 40$  м), скорость передачи сообщений может достигать до 1 Мбит/с. Более длинные сетевые расстояния уменьшают доступную скорость передачи, например до 125 Кбит/с в сети длиной до 500м. Высокоскоростной CAN (“**High speed**” **CAN**) сетью, считается сеть со скоростью передачи данных более 500 Кбит/с.

## Основные принципы CAN

Детали спецификации **CAN** протокола полностью описаны в **Robert Bosch GmbH, “CAN Specification 2.0,” 1991**. Ознакомиться с документом в формате **PDF** можно по следующему адресу <http://esd.cs.ucr.edu/webres/can20.pdf>. Далее я дам максимально возможно краткое описание того как данные передаются по **CAN**, как структурированы сообщения **CAN** и как обрабатываются ошибки передачи сообщений.

Есть четыре типа сообщений **CAN**, или фреймы (**frames**): фрейм данных (**Data Frame**), удаленный фрейм (**Remote Frame**), ошибочный фрейм (**Error Frame**) и фрейм перегрузки (**Overload Frame**).

**Data Frame** — стандартное сообщение CAN, широкопередатчельно передаваемые данные от передатчика на другие узлы сети.

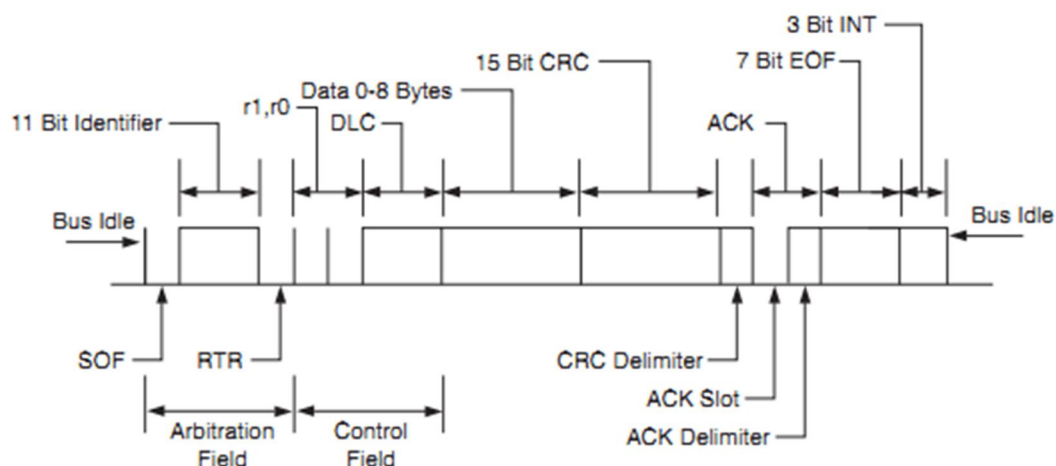
**Remote Frame** — широкопередатчельно передаваемое передатчиком сообщение, на запрос данных от конкретного узла сети.

**Error Frame** — может быть передан любым узлом, который обнаруживает ошибку в сети.

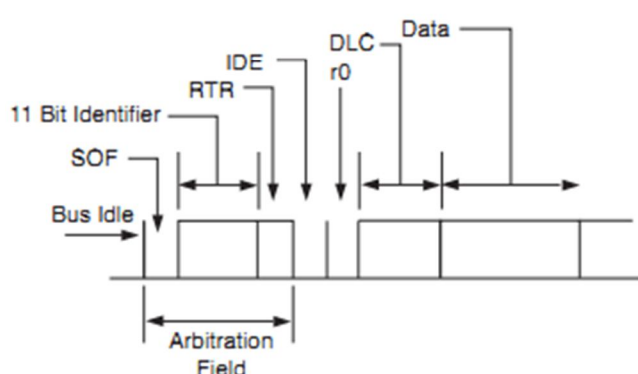
**Overload Frame** — используются как запрос на предоставление дополнительной паузы между получаемыми данными (**Data Frame**) или запросами на получение данных (**Remote Frame**).

Ниже проиллюстрированы различия между **Data Frames** для стандартов **CAN 2.0A** и **CAN 2.0B**, - **рис. 2**

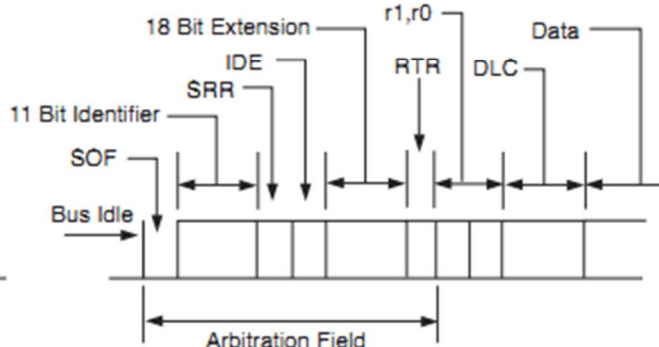
### CAN 2.0A Message Frame



### CAN 2.0B Message Frame (Standard Format)



### CAN 2.0B Message Frame (Extended Format)



Различие между форматами **CAN 2.0A** и **CAN 2.0B** заключается в том что фрейм данных для **CAN 2.0B** поддерживает как стандартный идентификатор фрейма данных — 11 бит, так и расширенный идентификатор фрейма данных — о 29 бит. Фреймы стандартного и расширенного формата могут без проблем передаваться по одной на той же шине, и даже иметь в цифровой форме эквивалентный идентификатор. В этом случае у стандартного фрейма будет более высокий приоритет,- **рис. 3**



Table 1: CAN 2.0A Message Frame		
Field	Length (bits)	Description
Start of Frame (SOF)	1	Must be dominant
Identifier	11	Unique identifier indicates priority
Remote Transmission Request (RTR)	1	Dominant in data frames; recessive in remote frames
Reserved	2	Must be dominant
Data Length Code (DLC)	4	Number of data bytes (0–8)
Data Field	0–8 bytes	Length determined by DLC field
Cyclic Redundancy Check (CRC)	15	
CRC Delimiter	1	Must be recessive
Acknowledge (ACK)	1	Transmitter sends recessive; receiver asserts dominant
ACK Delimiter	1	Must be recessive
End of Frame (EOF)	7	Must be recessive

### Описание фрейма сообщения стандарта **CAN 2.0A**

Начало сообщения (**Start of Frame (SOF)**) — 1 бит, должен быть доминантным.

Идентификатор (**Identifier**) — 11 бит, уникальный идентификатор, указывает приоритет.

Удаленный запрос на передачу (**Remote Transmission Request (RTR)**) — 1 бит, доминантный в сообщении и рецессивный в запросе на передачу сообщения.

Резерв (**Reserved**) — 2 бита, должны быть доминантными.

Длина кода данных (**Data Length Code (DLC)**) — 4 бита, количество байтов данных (0-8).

Поле передаваемых данных (**Data Field**) — от 0 до 8 байт, размер определен в поле **DLC**.

Контрольный циклический избыточный код (**Cyclic Redundancy Check (CRC)**) — 15 бит.

Разделитель **CRC** — 1 бит, должен быть рецессивный.

Подтверждение (**Acknowledge (ACK)**) — 1 бит, передатчик отправляет рецессивный; получатель подтверждает доминантным.

Разделитель **ACK** — 1 бит, должен быть рецессивным.

Завершение сообщения (**End of Frame (EOF)**) — 7 бит, должны быть рецессивными, - **рис. 4**

Table 2: CAN 2.0B Message Frame		
Field	Length (bits)	Description
Start of Frame (SOF)	1	Must be dominant
Identifier – Standard and Extended Formats	11	Unique identifier corresponds to Base ID in Extended Format
Identifier – Extended Format	29	Comprised of 11 bit Base ID and 18 bit Extended ID
Remote Transmission Request (RTR) – Standard and Extended Formats	1	Dominant in data frames; recessive in remote frames. In Standard Format, the 11 bit identifier is followed by the RTR bit.
Substitute Remote Request (SRR) – Extended Format	1	Must be recessive. SRR is transmitted in Extended Frames at the position of the RTR bit in Standard Frames. In arbitration between standard and extended frames, recessive SRR guarantees the standard message frame prevails.
IDE – Standard and Extended Frames	1	Must be recessive for Extended Format; dominant for Standard Format.
Reserved r0 – Standard Format	1	Must be dominant
Reserved r1, r0 – Extended Format	2	Must be recessive
Data Length Code (DLC)	4	Number of data bytes (0–8)
Data Field	0–8 bytes	Length determined by DLC field
Cyclic Redundancy Check (CRC)	15	
CRC Delimiter	1	Must be recessive
Acknowledge (ACK)	1	Transmitter sends recessive; receiver asserts dominant
ACK Delimiter	1	Must be recessive
End of Frame (EOF)	7	Must be recessive

### Описание фрейма сообщения стандарта **CAN 2.0B**

Начало сообщения (**Start of Frame (SOF)**) — 1 бит, должен быть доминантным.

Идентификатор стандартного и расширенного форматов (**Identifier**) — 11 бит, уникальный идентификатор, соответствует базовому **ID** в расширенном формате.

Идентификатор расширенного формата (**Identifier – Extended Format**) — 29 бит, состоит из 11 бит базового **ID** и 18 бит расширенного **ID**.

Удаленный запрос на передачу (**Remote Transmission Request (RTR)**) стандартный и расширенный форматы — 1 бит, доминантный в сообщении и рецессивный в запросе на передачу сообщения. В стандартном формате 11 бит идентификатора следуют за битом **RTR**.

Замещение удалённого запроса (**Substitute Remote Request (SRR)**). Для расширенного формата — 1 бит, должен быть рецессивный. **SRR** передаются в расширенных форматах сообщений на позиции бита **RTR** в стандартном сообщении. В арбитраже между стандартными и расширенными сообщениями, рецессивные **SRR** обеспечивает приоритет стандартным сообщениям.

Поле **IDE** — для стандартного и расширенного форматов — 1 бит, должен быть рецессивным для расширенного формата и доминантным для стандартного.

Резерв (**Reserved r0**) для стандартного формата — 1 бит, должен быть доминантным.

\*\*\*\*\*

[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей



Резерв (**Reserved r1, r0**) для расширенного формата — 2 бита, должны быть рецессивными.

Длина кода данных (**Data Length Code (DLC)**) — 4 бита, количество байтов данных (0-8).

Поле передаваемых данных (**Data Field**) — от 0 до 8 байт, размер определен в поле **DLC**.

Контрольный циклический избыточный код (**Cyclic Redundancy Check (CRC)**) — 15 бит.

Разделитель **CRC** — 1 бит, должен быть рецессивный.

Подтверждение (**Acknowledge (ACK)**) — 1 бит, передатчик отправляет рецессивный; получатель подтверждает доминантным.

Разделитель **ACK** — 1 бит, должен быть рецессивным.

Завершение сообщения (**End of Frame (EOF)**) — 7 бит, должны быть рецессивными.

## Фрейм данных CAN

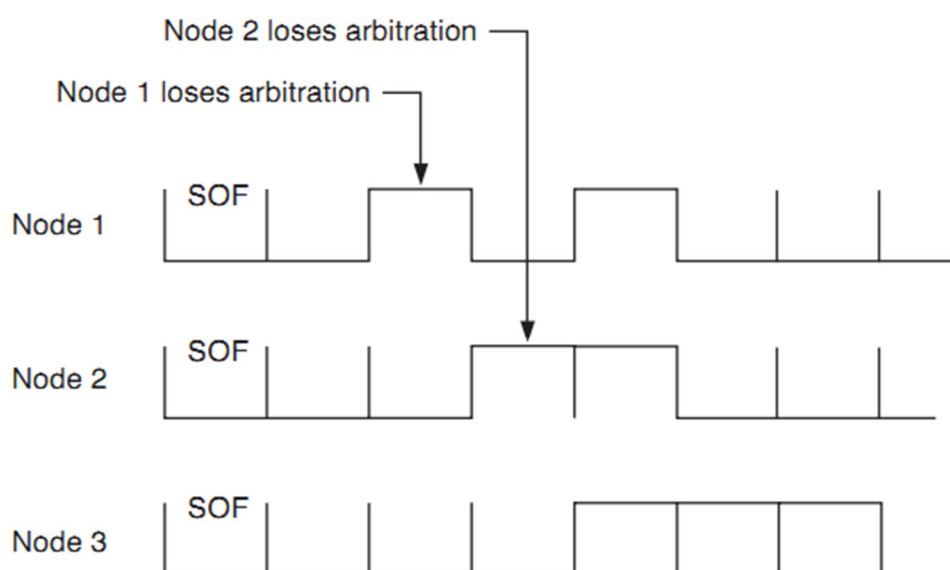
Фрейм данных **CAN** состоит из семи полей: Начало фрейма (**SOF**), арбитраж, управление, данные, циклические, проверка по избыточности (**CRC**), подтверждение (**ACK**) и конец фрейма (**EOF**). Биты сообщения **CAN** обозначены как "доминирующие" (0) или "рецессивные" (1). Поле **SOF** состоит из одного доминирующего бита. Все сетевые узлы синхронно ожидают команды разрешения на передачу сообщений и начинают передавать одновременно. Арбитражная схема определяет, какой из узлов, пытающихся передавать сообщения имеет главный приоритет и фактически будет управлять шиной.

## Арбитраж (Arbitration)

Арбитражное поле сообщения **CAN** состоит из 11-или 29-разрядного идентификатора и бита удаленной передачи (**RTR**). Арбитражную схему **CAN** называют "*носителем контроля с множественным доступом и обнаружением коллизий*" или **CSMA/CD**, которая гарантирует, что **самое важное сообщение с наивысшим приоритетом будет передано по всей сети в первую очередь**. Приоритет сообщения определен численным значением идентификатора в

\*\*\*\*\*  
[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

арбитражном поле, поле с самым низким численным значением имеет самый высокий приоритет. Неразрушающий, интеллектуальный арбитраж разрешает конфликты среди конкурирующих передатчиков. Это означает, что шина может считаться действующей как логический элемент И (**AND gate**). Если какой-либо узел пишет по сети доминантный признак (0), то каждый узел читает доминирующий бит независимо от его назначения, заданного передающим узлом. Каждый передающий узел всегда читает ответ на каждый переданный бит. Если узел передает рецессивный бит запроса на отправку сообщения и получает доминирующий бит для прочтения сообщения, он сразу же прекращает передавать. Ниже проиллюстрирован приоритет сетевого арбитража где третий узел имеет высший приоритет и первый низший, - **рис. 5**



CAN Arbitration: Node 3 has highest, and Node 1 the lowest, priority messages.

Бит **RTR** включён для того чтобы различать сообщения для передачи и удаленные запросы на приём сообщений. В стандартных сообщениях для передачи (**Data Frame**) бит **RTR** должен быть доминантным, а в удаленных запросах на приём сообщений (**Remote Frame**) должен быть рецессивным.

### Контрольное поле и поле данных в сообщении (Control and Data Fields)

Поле управляющее длиной кода данных (**DLC**) состоит из 6 бит (из которых используются только 4 младших бита), они показывают количество данных в

\*\*\*\*\*  
[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

сообщении. Поскольку только до 8 байт данных может быть отправлено в одном сообщении, поле **DLC** может принимать значения в диапазоне от 000000 до 000111. Данные которые должны быть переданы содержатся только в поле данных. В первую очередь передается наиболее значимый байт (**Most significant byte (MSB)**) из байтов данных.

### Обработка ошибок (Error Handling)

В протоколе **CAN** реализовано пять уровней обнаружения ошибок. На уровне сообщений, выполняется циклическая проверка избыточности (**Cyclic Redundancy Check (CRC)**), проверки сообщения и обязательное подтверждение проверок (**Acknowledge (ACK)**). Бит проверки уровней состоит из монитора и наполнения.

Циклические ошибки избыточности обнаруживаются, используя код **CRC** размером 15 битов, вычисленный передатчиком из контента сообщения. Каждый получатель, принимающий сообщение, повторно вычисляет код **CRC** и сравнивает его с переданным значением. Несоответствие между этими двумя вычислениями заставляет установить флаг (**flag**) ошибки. Проверяемые сообщения, в которых будет установлен флаг ошибки, это обнаружение получателем недопустимого бита в разделителе **CRC**, разделителе **ACK**, в завершении сообщения **EOF** или в 3-х битном разделяющем сообщения пространстве. В конечном итоге каждый принимающий узел записывает доминантный бит в ячейку разделителя **ACK**, которая затем читается отправившим это сообщение узлом. И если приём сообщения получателем не подтверждён (возможно потому что получающий узел перестал работать) то устанавливается флаг ошибки подтверждения (**ACK**).

На уровне битов мы уже отметили, что каждый переданный бит считывается снова передатчиком этого сообщения при контроле подтверждения о получении сообщения, присланного получателем. Если контролируемое значение отличается от отправленного, значит на уровне битов обнаружена ошибка. Дополнительно, ошибки на уровне битов обнаруживаются при помощи «вставок»: После пяти последовательных идентичных битов, которые передаются в сообщении следует «вставка», бит противоположной полярности вставляется передатчиком в поток передаваемых битов (биты «вставки» вставляются в сообщение от поля **SOF** до поля **CRC**). Получатели автоматически проверяют сообщение на наличие «вставок». Если любой из принимающих узлов сети обнаруживает в полученном сообщении шесть последовательных идентичных битов, то фиксируется ошибка (отсутствия «вставки»). В дополнение для обнаружения ошибок, «вставки» гарантируют, что есть достаточно не нулевых окончаний в потоке битов (**non-return to zero (NRZ)**), чтобы поддержать синхронизацию.

\*\*\*\*\*  
[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

## Сообщение об ошибке (The CAN Error Frame)

Если передающий или принимающий узел обнаруживает ошибку, он немедленно прерывает приём или передачу текущего сообщения. Сообщение об ошибке называемое «флаг» ошибки, состоит из шести доминантных битов и разделителя сообщения об ошибке состоящего из восьми рецессивных битов. Так как эта строка битов нарушает правило «вставок», все другие узлы также передают сообщение об ошибке. После критического количества обнаруженных ошибок, узел в конце концов само-отключается. Надежность, особенно в производстве и автомобильной электронике, где применяется технология CAN, требует, чтобы сеть могла отделять случайные ошибки (из-за скачков напряжения, шумов или других временных причин) от постоянных, являющихся причиной неисправности узла из-за дефектов в оборудовании. Следовательно, узлы хранят и отслеживают число обнаруженных ошибок. Узел может быть в одном из трех режимов в зависимости от количества зафиксированных ошибок:

Если количество зафиксированных ошибок в каждом буфере передачи или приёма соответствующего узла, больше чем нуль и меньше чем 128, узел считается «активным с ошибкой» (**“error active”**), указывая на то, что несмотря на то, что узел остается полностью функциональным, по крайней мере одна ошибка была обнаружена.

Если количество зафиксированных ошибок между 128 и 255, то узел переходит в «пассивный с ошибками» (**“error passive”**) режим. В «пассивном с ошибками» режиме узел будет передавать на более медленном уровне, отправляя 8 рецессивных битов прежде чем снова отправить сообщение, распознав что шина свободна.

Если количество зафиксированных ошибок более 255, то узел переходит в режим «отключен от сети» (**“bus off”**), отключив себя самостоятельно.

Ошибка при получении добавляет в общее количество учтенных ошибок 1, ошибка при передаче добавляет в общее количество учтенных ошибок 8. Последующие безошибочные сообщения постепенно уменьшают количество учтенных ошибок на 1, за каждое безошибочное сообщение. Если общее количество учтенных ошибок возвращается к нулю, узел возвращается в нормальный режим функционирования. Узел в находящийся режиме **“bus off”** может перейти в режим **“error active”** после 128 входов в сеть из 11 последовательных рецессивных битов, которые были проконтролированы. Сообщение считается безошибочным, если передающий узел не нашёл в нём ошибок вплоть до поля **EOF**. Повреждённые сообщения отсылаются повторно сразу как только шина становится свободной.

## Запрос данных от конкретного узла сети (The CAN Remote Frame)

Узел, которому необходимы данные от другого узла сети, может запросить передачу таких данных, отправив соответствующий запрос на получение данных (**Remote Frame**). Например, микропроцессору управления центральным замком на вашем автомобиле необходимо знать положение селектора коробки передач от ЭБУ трансмиссии (является ли он в положении «паркинг»). Структура запроса на получение данных аналогична структуре стандартного сообщения только без поля данных (**data field**) и с рецессивным **RTR** битом.

## Запрос на предоставление дополнительной паузы между получаемыми данными и свободное пространство между сообщениями (Overload Frames and Interframe Space)

Если какой-либо узел сети будет получать сообщения быстрее, чем он может их обработать, то будет сгенерирован запрос на предоставление дополнительной паузы между получаемыми данными (**Overload Frames**) чтобы обеспечить дополнительное время между принимаемыми данными или запросами на получение сообщений (**Remote Frame**). Подобно сообщению об ошибке, **Overload Frame** имеет два поля с битами: **flag** перегрузки состоящий из шести доминирующих битов и разделитель перегрузки, состоящий из восьми рецессивных битов. В отличие от сообщений об ошибке, суммарный подсчёт **Overload Frames** не ведётся.

Пространство между сообщениями состоит из трех рецессивных битов так же, как и время простоя шины между сообщениями или удаленными запросами на передачу. Во время перерыва никакому узлу не разрешают инициировать передачу (если доминирующий бит будет обнаружен во время Перерыва, то **Overload Frame** будет сгенерирован). Время простоя шины длится, пока у узла нет чего-то для передачи, а когда начинается передача, то в этот момент появление доминирующего бита на шине сигнализирует о начале передачи сообщения **SOF**.

## Загрузка шины (Bus Loading)

**CAN** обеспечивает устойчивое, простое и гибкое сетевое решение для производственных, автомобильных и многих других приложений. **Главный недостаток протокола CAN** - что задержка сообщения не является определённой (из-за существования **Error Frames**, **Overload Frames** и повторных передач), и увеличения задержки ведёт к увеличению сетевого трафика. В целом использование шины не должно превышать 30% от максимальной мощности

\*\*\*\*\*  
[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей



шины и гарантировать, что низкоприоритетные сообщения не испытывают недопустимую задержку. Использование шины определено как деление двух величин — общее количество использованных для передачи битов поделённое на общее максимально доступное количество для передачи битов, и вычисляется следующим образом:

Шаг 1 — Выбирается единица времени  $\geq$  самого медленного зафиксированного периодического сообщения в сети (обычно 1 секунда).

Шаг 2 — Определяются все периодические сообщения.

Шаг 3 — К каждому из этих сообщений приблизительно одинакового размера, добавляются 47 служебных бит (размер служебных полей данных - SOF + Arbitration + RTR + Control + CRC + Acknowledgment + EOF +

Interframe Space =  $1 + 11 + 1 + 6 + 16 + 2 + 7 + 3 = 47$  bits).

Шаг 4 — Рассчитывается количество бит используемых в сообщениях путем умножения размера сообщения в битах на количество передач выполняемых в единицу времени.

Шаг 5 — Суммирование всех битов используемых в переданных сообщениях для оценки общего объёма сетевого трафика. Умножение этой величины на страховочный коэффициент 1.1 для получения наихудшего прогноза сетевого трафика.

Шаг 6 — В завершении, поделите общее количество использованных для передачи битов на общее максимально доступное количество для передачи битов (например, 125 Кбит/с или 500 Кбит/с умножаются на единицу времени) для получения предполагаемого процента загрузки шины, - **рис. 6**

Table 2: Bus Loading Example				
Message	Data (bytes)	Message Size (bits)	Rate and Period	Message Bits Consumed
MsgA	0	47	10 trx/s: 100 ms	$10 \cdot 47 = 470$ bps
MsgB	5	$5 \cdot 8 + 47 = 87$	2 trx/s: 500 ms	$87 \cdot 2 = 174$ bps
MsgC	8	$8 \cdot 8 + 47 = 111$	1 trx/s: 1 s	$111 \cdot 1 = 111$ bps
...	...	...	...	...
			Total periodic bits consumed	10000 bps
Total Bits Consumed = $1.1 \cdot (\text{Total Periodic Bits Consumed}) = 11000$ bps				
Bandwidth Consumption = $11000/125000 = 8.8\%$				

## Протоколы синхронизированные по времени (Time-triggered Protocols)

Для контроля над сетью в реальном времени было бы желательно реализовать такой протокол связи, который гарантирует, что для сообщений

\*\*\*\*\*

[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

выбираются крайние временные параметры независимо от нагрузки на шину. Пример такого протокола, который контролирует временной уровень связи **CAN** данных, это "**time-triggered CAN**," или **TTCAN (ISO 11898-4)**. **TTCAN** сообщения имеют два специальных типа, называемые «окна времени» (**time windows**): привилегированные окна времени (**exclusive time windows**), и арбитражные окна времени (**arbitrating time windows**). **Exclusive time windows** прикреплены к специальным сообщениям, которые передаются периодически. Таким образом, **Exclusive time windows** не конкурируют за доступ к шине. **Arbitrating time windows** используются для сообщений не имеющих строгих ограничений по времени.

**Arbitrating time windows**, как нормальные сообщения **CAN**, конкурируют за доступ к шине на основе приоритета через арбитраж. **Time-triggered CAN** протокол, требует наличия в сети "главного узла" (**master node**), который периодически широковещательно передает сигнал времени сети (называемый глобальным временем (**global time**)) в специальном информационном сообщении. Для повышения отказоустойчивости в сети должны быть несколько потенциальных главных узлов. Если главный узел перестал работать (обнаружено отсутствие специального информационного сообщения), другие потенциальные главные узлы конкурируют за статус «главного узла» при помощи арбитража и новым «главным узлом» становится узел с самым высоким приоритетом. После этого новый главный узел начинает широковещательно передавать специальные информационные сообщения — **global time**. **Time-triggered CAN** протокол не ретранслирует повреждённые сообщения, и при этом это не вызывает Error Frames.

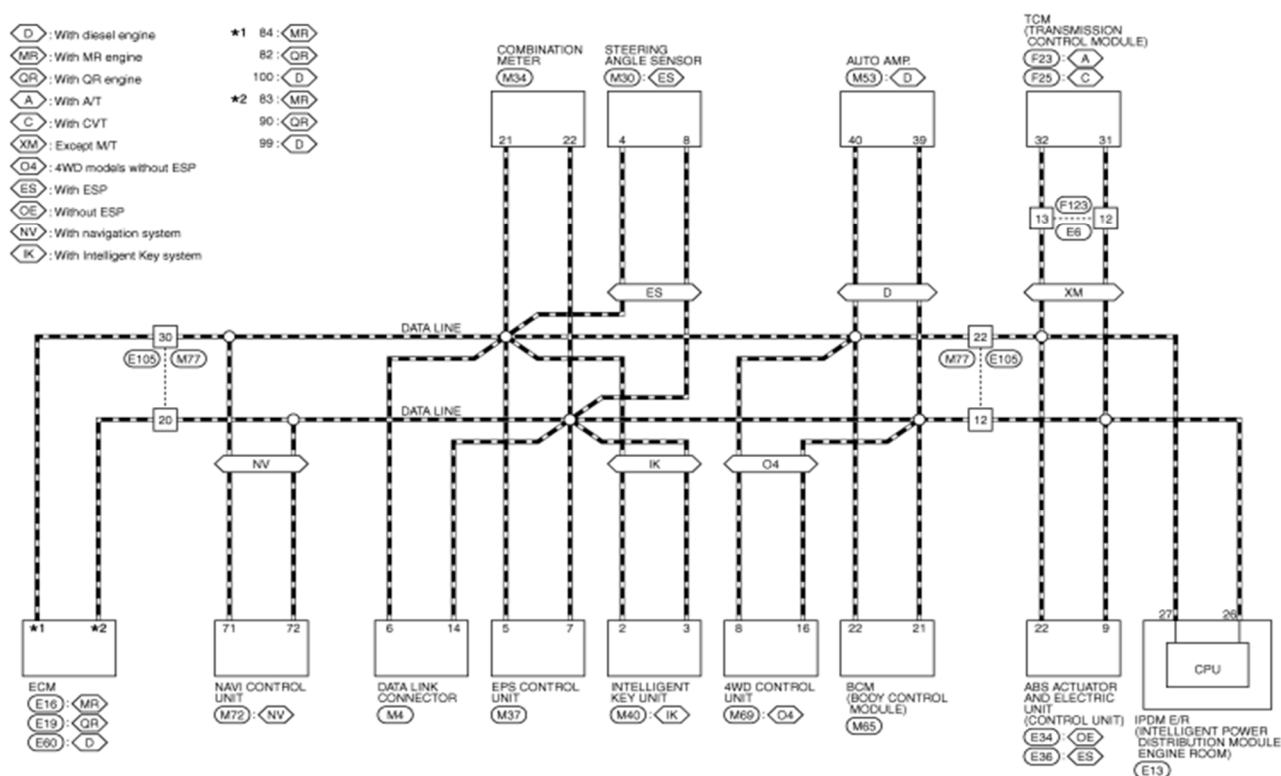
У протокола **TTCAN** есть конкурирующий протокол **FlexRay**, разработанный консорциумом автомобильных производителей и поставщиков. Коммуникационное сообщение (фрейм) **FlexRay** состоит из периодических синхронизированных "статических" и "динамических" частей. Статический сегмент составлен из одинаковой длины временных интервалов, соответствующих соединенным в сеть узлам. Каждый узел передает свои сообщения синхронно в его зарезервированном слоте. Статический сегмент также передает "синхронизирующий" кадр, чтобы обеспечить глобальную переменную (**timebase**) для сети. В отличие от **CAN**, нет никакого арбитража для шины. Динамический сегмент - по существу механизм "опроса" где каждому узлу дают возможность поместить инициированное событие или асинхронное сообщение в шину в порядке приоритетов, используя механизм синхронизации «миниразделения на слоты». Для повышения отказоустойчивости, узлы сети использующей протокол **FlexRay**, могут быть связаны двумя шинами или каналами одновременно.

Ну вот, в принципе, вся основная информация о протоколе **CAN**, а теперь немного о том, **как выглядит CAN шина на примере автомобилей**

\*\*\*\*\*  
[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

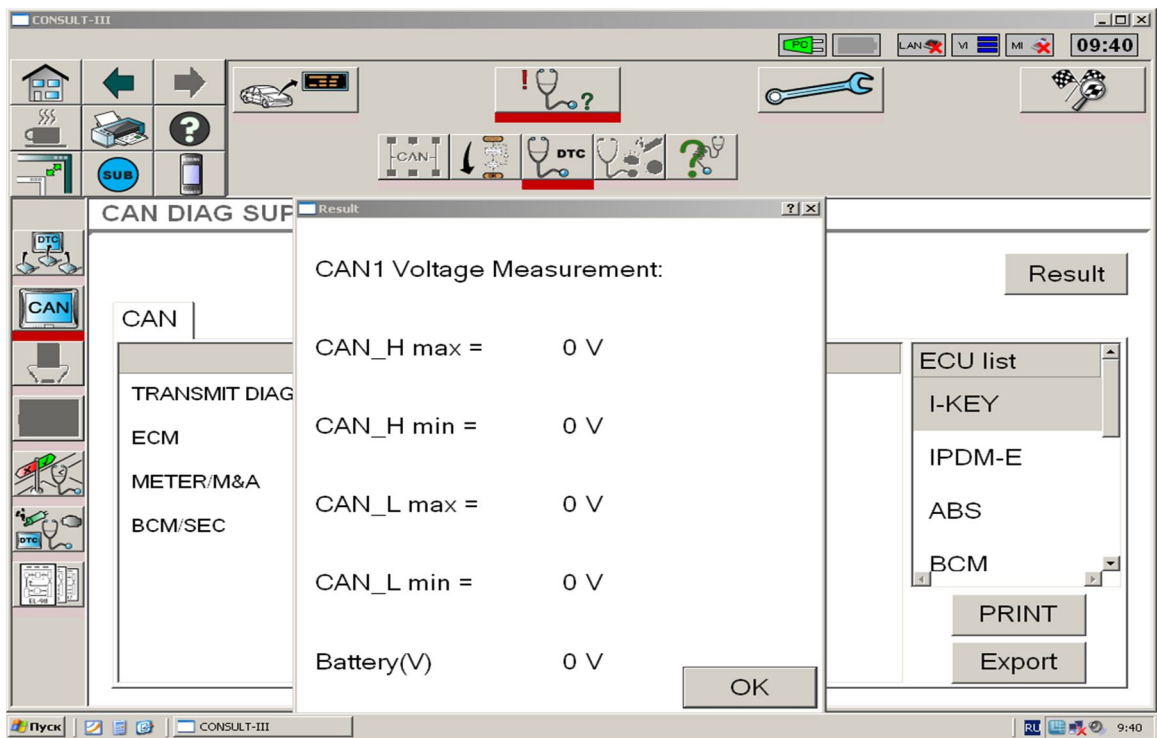
**произведённых в Японии.** Сразу хочу отметить, что без надлежащего диагностического оборудования проводить диагностику и ремонт неисправностей **CAN** шины можно в очень ограниченном диапазоне. Всё сведётся к проверке физической целостности проводов, проверки состояния соединительных разъёмов, проверки сопротивления проводки и **Terminal resistor**, проверки соответствующего уровня напряжения на **CAN low** и **CAN high** линиях. Применение в диагностике дилерского оборудования тоже лишь облегчит проверку и сузит круг поиска неисправности, с очень большой неохотой автопроизводители допускают контакт с программным обеспечением, своей интеллектуальной собственностью. В случае проблем на программном уровне возможно только перепрограммирование или замена соответствующего ЭБУ.

#### CAN SYSTEM (LHD MODELS)



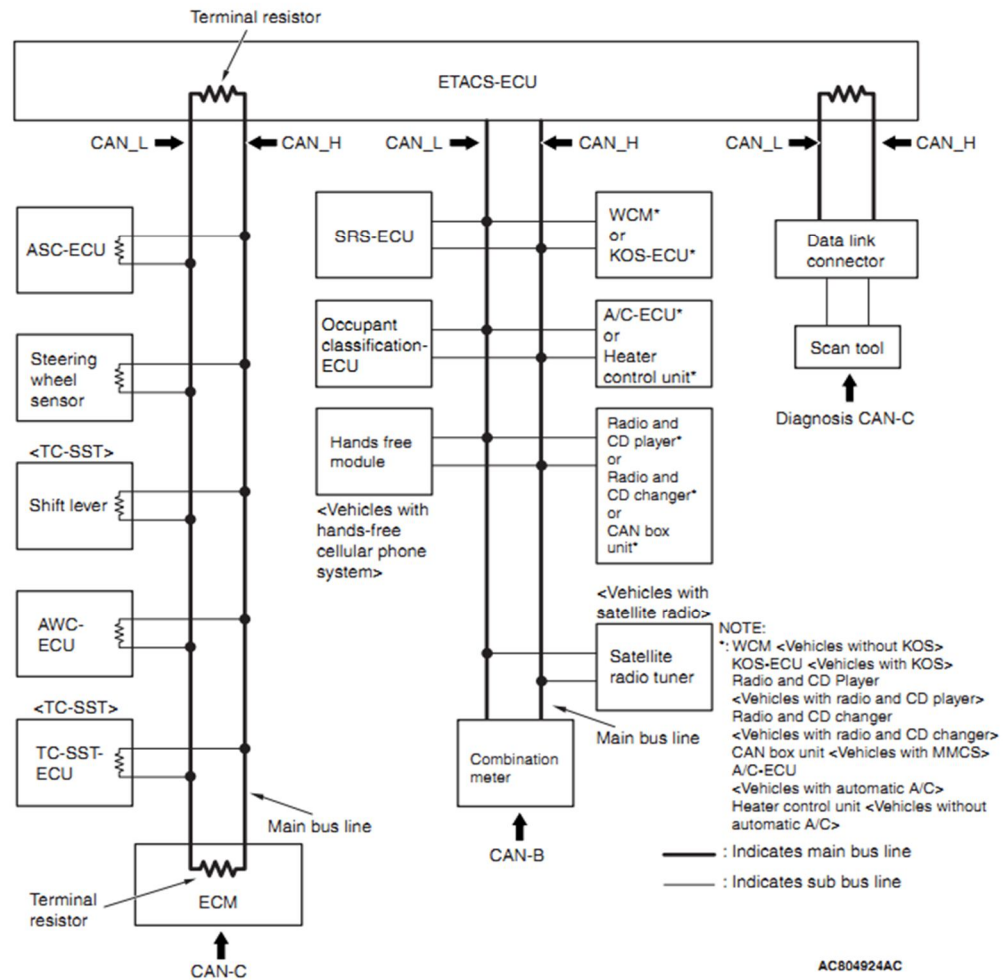
**Пример CAN шины автомобиля Nissan 2007г.в. - Рис. 7**

Интерфейс программы Consult III (Nissan), окно диагностики CAN шины, - **рис. 8**



Пример CAN шины автомобиля Mitsubishi 2004г.в. – *рис. 9*

STRUCTURE



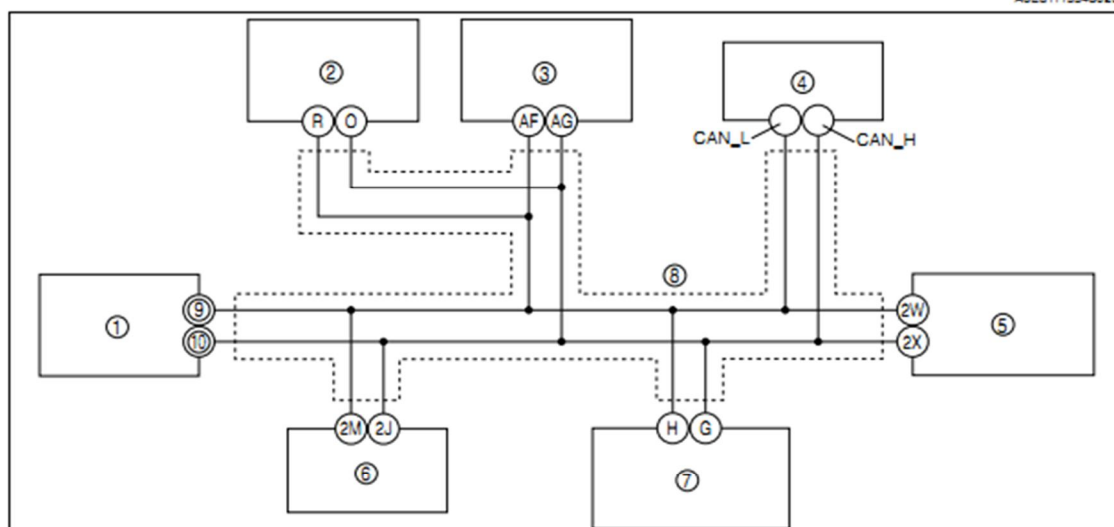
Пример CAN шины автомобиля Mazda 2004г.в. – *рис. 10*



## MULTIPLEX COMMUNICATION SYSTEM

### SYSTEM WIRING DIAGRAM

A5EB11155430203



A5EB1111102

1	PCM
2	DSC HU/CM (with DSC)
3	ABS (ABS/TCS) HU/CM (with ABS (ABS/TCS))
4	Data link connector-2
5	Instrument cluster
6	TCM (JA5AX-EL)

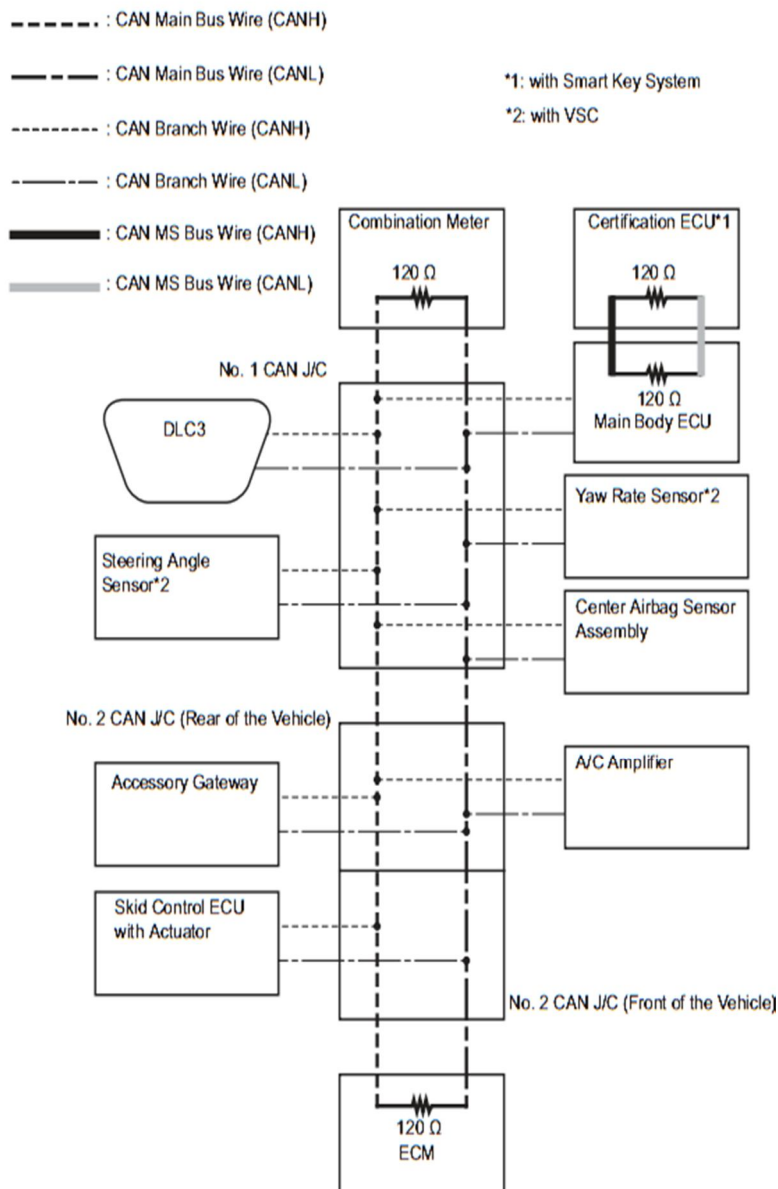
7	4WD control module (4WD)
8	Twist pair
9	2R (except MZR-CD (RF Turbo))
	39 (MZR-CD (RF Turbo))
10	2U (except MZR-CD (RF Turbo))
	13 (MZR-CD (RF Turbo))

Пример **CAN** шины автомобиля **Toyota** 2007г.в. – **рис. 11**

\*\*\*\*\*

[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

## SYSTEM DIAGRAM



За дополнительной и полной информацией по системе каждого диагностируемого автомобиля следует обращаться к соответствующим документам автопроизводителей (**WSM** и **TSB**).

Удачных всем ремонтов и беспроblemного обслуживания своих автомобилей.

**Боровиков Игорь Александрович**

(ник на форуме Легион-Автодата <http://forum.autodata.ru/index.php> – semirek)

\*\*\*\*\*

[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей

## **СОЮЗ АВТОМОБИЛЬНЫХ ДИАГНОСТОВ**

Автосервис "Япония Авто"

г. Калининград, ул. Портовая, 45

+7 [4012] 63 12 55, 65 60 99, +7(911) 475 9493

[http://www.japanauto.ru /](http://www.japanauto.ru/)

\*\*\*\*\*  
[www.autodata-online.ru](http://www.autodata-online.ru) - База данных по ремонту и диагностике автомобилей  
[www.motordata.ru](http://www.motordata.ru) - Интерактивная база данных по диагностике автомобилей  
[www.autodata.ru](http://www.autodata.ru) - Интернет-магазин литературы по ремонту автомобилей